

Physics of Language Models: Part 2.2, How to **Learn From Mistakes** on Grade-School Math Problems

Result 1

LLM often “knows” it has made reasoning mistakes

internal states can appear “regretful” (it wants to backspace!)
but letting it retry with randomness is not very efficient

Result 2-5

pretrain with error + immediate corrections \Rightarrow acc boost

more errors the merrier \Rightarrow accuracy 78% \rightarrow 94% on iGSM data
model learns to correct errors on-the-fly (no multi-stage prompting)

Result 6-7

beam-search / fine-tune \Rightarrow no significant accuracy gain

to learn to correct errors, must pretrain a model with mistakes

Result 8

pretrain with fake mistakes \Rightarrow no more regret!

generate fake mistakes for free \Rightarrow acc 78% \rightarrow 91% on iGSM data


Even the strongest LLMs still make reasoning mistakes

User:


Each box Y has a box Q inside it. Each box Q has a box A inside it. Each box Q has a box O inside it. Each box L has a box K inside it. Each box Q weights 0 pounds on its own. ... Each box L weights 2 pounds on its own. Each box O weights 3 pounds on its own. ... Each box X has a box Y inside it. **What is the total weight of box X?**

GPT-4o:

To determine the total weight of box X, we need to calculate the weight of box X and the weight of the innermost boxes and working our way out.

1. ****Box L****: Weighs 2 pounds.  (missing box K inside L)
2. ****Box K****: Weighs 0 pounds on its own, but ...

... ..

8. ****Box X****: Weighs 6 pounds on its own, contains box Y.
- Total weight of box X = 6 + 74 (box Y) = 80 pounds.
Therefore, the total weight of box X is 80 pounds. 

If model can correct mistakes via **multi-round prompting**, why can't it correct the mistake **immediately after**?

User:

Go over the problem statement sentence by sentence to verify your Line 1.

GPT-4o:

Certainly! Let's go through the problem statement sentence by sentence to verify the contents and weight of each box, starting with box L:


... ..

4. ****"Each box L has a box K inside it."**
- This tells us that box L contains box K. 

... ..

14. ****"Each box L weights 2 pounds on its own."**

... ..


My initial calculation for the weight of box L was correct. 

User:

Verify your answer. This is not correct.

GPT-4o:

Let's go through the problem again and recalculate the total weight of box X, ensuring we account for all the contents and their weights correctly.

1. ****Box L****: Weighs 2 pounds on its own and contains a box K. 
 - ****Box K****: Weighs 0 pounds on its own and contains a box R.
 - ****Box R****: Weighs 8 pounds on its own.
 - Total weight of box K = 0 + 8 = 8 pounds.
 - Total weight of box L = 2 + 8 = 10 pounds.

... ..

Therefore, the total weight of box X is 120 pounds. 



Correcting mistakes during generation (right after the mistake) could avoid generating useless tokens following an erroneous step, and speed up the inference

So, what's the best way to achieve this?

LLMs make mistakes in the middle of CoT reasoning steps

1. Define Dance Studio's Messenger Bag as a ; so $a = 3$.
2. Define Lakeshore High's Dance Studio as b ; so $b = 2$.
3. Define Lincoln High's Dance Studio as c ; so $c = a * 7 = 21$.
4. Define Dance Studio's Canvas Backpack as d ; **error starts here**

Example from the iGSM data created in Part 2.1.

Explanation: model wants to compute a quantity, but after writing it done, model realizes that this quantity is not ready for computation.



on position after a mistake is made, *probing* reveals that **model sometimes “knows” it has made a mistake**

⇒ internal states exhibit a “regretful” pattern

⇒ a pretrained LLM (on error-free data) can almost **already** be an error detector

⇒ can easily fine-tune for error detection

What if we let LLMs “retry upon regret”

backspace to previous step when it thinks it made a mistake + **regenerate from there**

accuracy 78% → 80% (on iGSM data)

(for comparison) 78% → 78% with beam search

Downside: accuracy improvement is small

after all, this only uses randomness to retry

need a highly-accurate error detector

99%-accurate detector only gives 2% improvement

100% detector gives 7% improvement, but that's too ideal

high inference cost

need error detector model + multiple regenerations

⇒ To get closer to AGI? Can we have a single model, autoregressive, no multi-round generations?

math data with retry

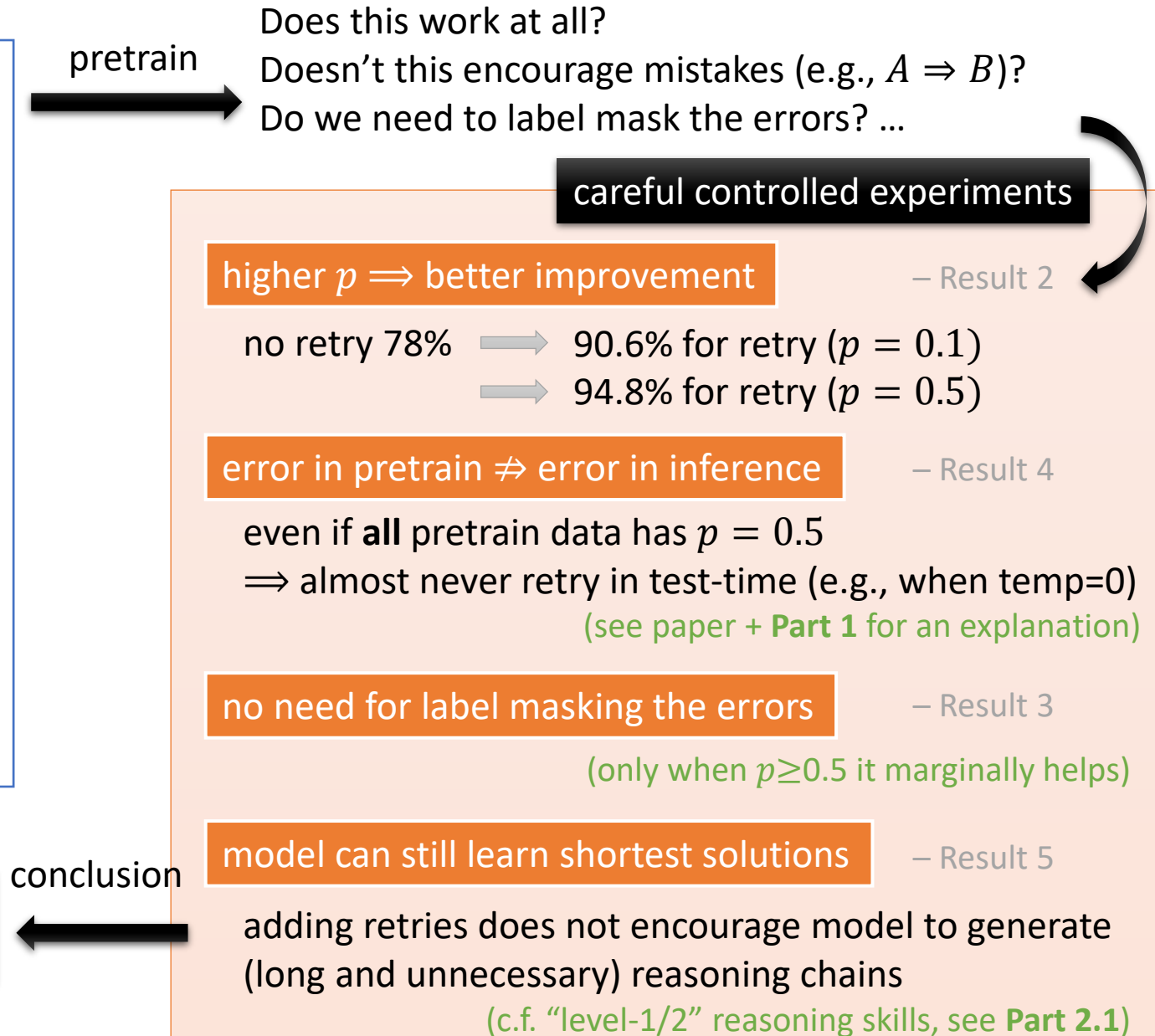
Concept: prepare *retry data* of the form
“ $A \Rightarrow B$, oh I made a mistake, actually $A \Rightarrow C$ ”

Let’s see how this works on the synthetic iGSM data:

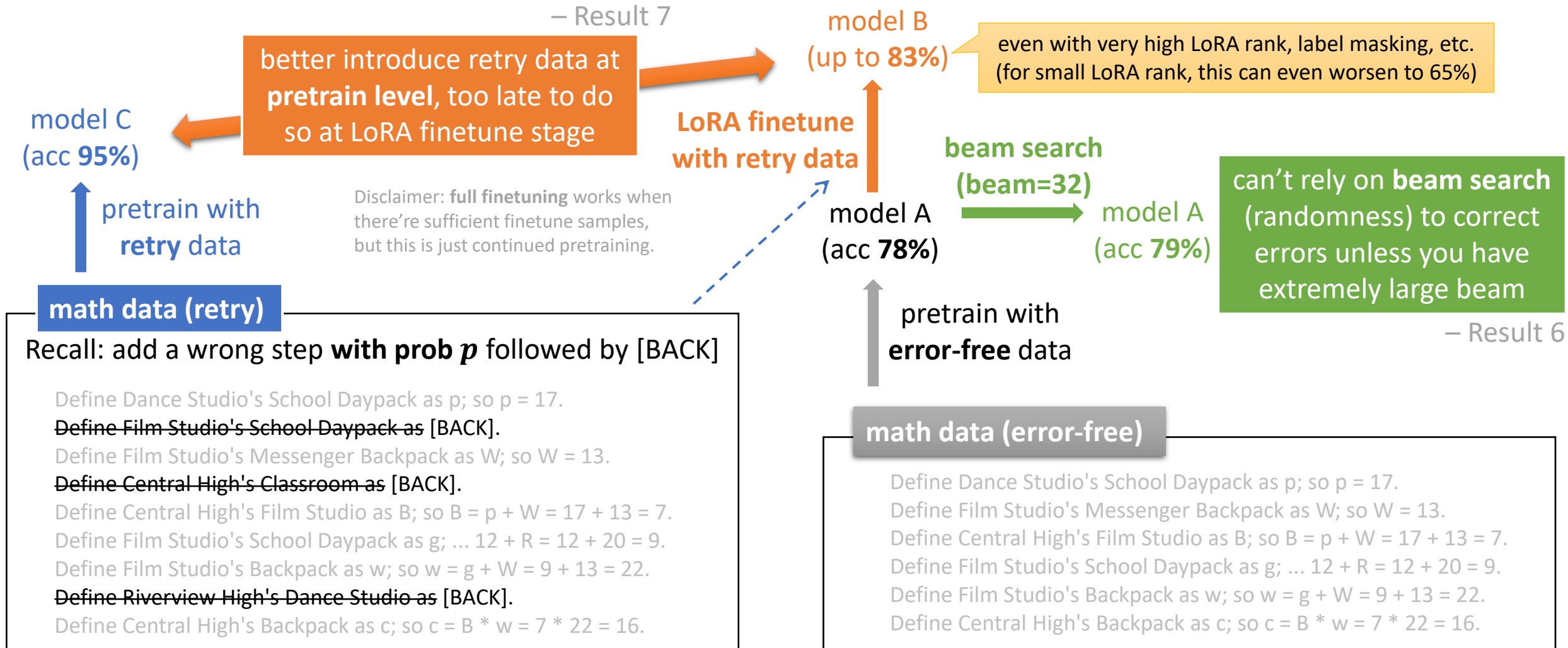
Define Dance Studio's School Daypack as p ; so $p = 17$.
Define Film Studio's School Daypack as [BACK].
Define Film Studio's Messenger Backpack as W ; so $W = 13$.
Define Central High's Classroom as [BACK].
Define Central High's Backpack as [BACK].
Define Central High's Film Studio as B ; so $B = p + W = 17 + 13 \dots$
Define Film Studio's School Daypack as g ; ... $12 + R = 12 + 20 \dots$
Define Film Studio's Backpack as w ; so $w = g + W = 9 + 13 = 22$.
Define Riverview High's Dance Studio as [BACK].
Define Central High's Backpack as c ; so $c = B * w = 7 * 22 = \dots$

add a wrong step with prob p followed by [BACK]

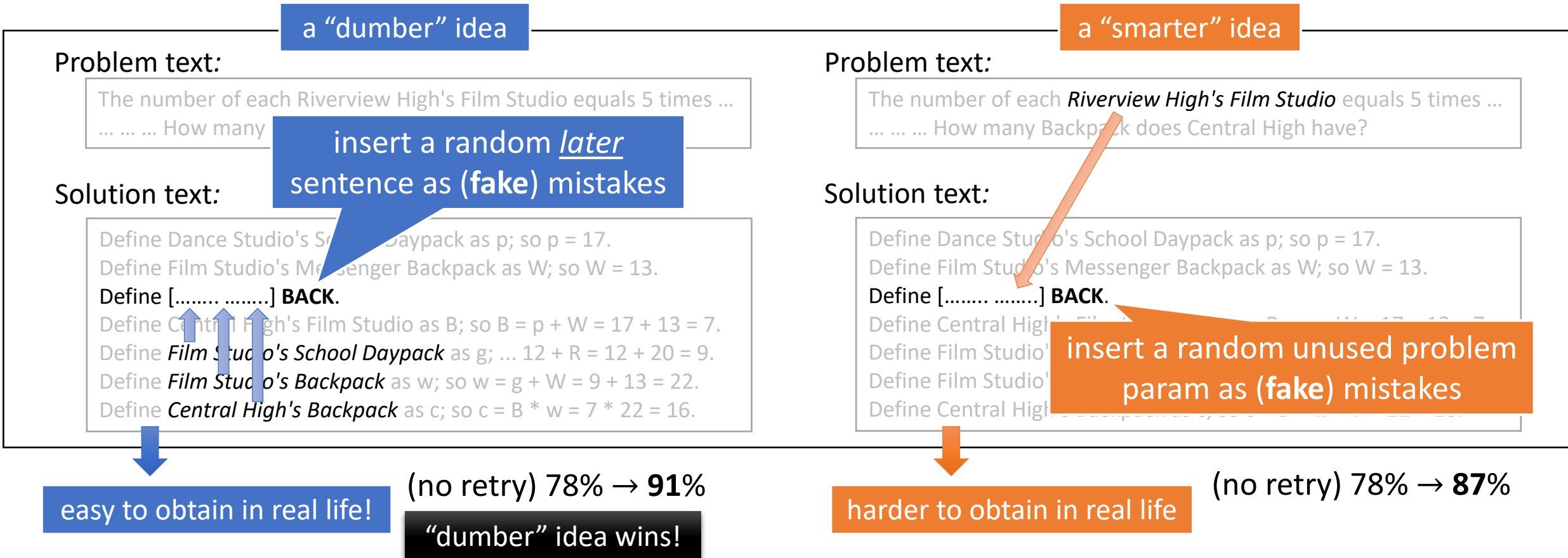
very safe to include math data with (lots of) errors and retries; **no change to pretrain/inference process**



unlike error detection (easy via finetune), **error correction** is a very different skill (i.e., substantial weight changes!) compared to error-free reasoning, potentially beyond what parameter-efficient fine-tuning (PEFT) can handle



How to prepare retry data (error + immediate corrections) in real life?



Future work:

We don't advocate for using this data to directly train future LLMs. However, since modern LLMs struggle with self-correction, should we consider adding synthetic math data with errors in similar fashions (e.g., prompting Llama3-405B to introduce errors)? We can't explore this due to GPU resource limits.